

Stop Negotiating, Start Collaborating

By David J. Anderson, June 2007

Stop Negotiating! No really! Cut it out! Banish it from your thoughts, your actions and your organization. Refuse to negotiate. When you need to work with another department, group or team inside your organization, and you find yourself in negotiations to get what you need, just say “Enough!” And refuse to negotiate.

What is so bad about negotiation?

Negotiation implies that you have an internal market and a market implies that you need a contract between a supplier and a consumer of a product or service (usually a service in an IT or technology product company context). Markets and their contracts carry transaction costs [1] – one of those costs is negotiation. If you can replace your internal market with a collaborative network, you’ll gain an economic boost from what are known as the network externalities [2] of being in the network. Collaborative networks do away with the transaction costs associated with marketplaces - specifically in this case, the cost of negotiation.

Negotiation also implies that information is being hidden. The supplier is hiding their availability, or the cost of the job, or some information that might give the consumer an *advantage*. Our 20th Century education taught us that “information is power” and business schools teach negotiation emphasizing that leverage involves hiding information. It’s become second nature in business to assume hidden information and to probe for it during negotiations. Hence, consumers of your IT service naturally distrust your estimates or resource requests.

Let’s pause for a moment and reflect on that word *advantage*. Within your organization, why would any one group or team require an *advantage* over another? Doesn’t the organization have a set of common goals? And isn’t the organization supposed to be collaborating to realize those goals? What can obtaining an *advantage* over another group possibly have to do with collaboration?

If you are beginning to think that there ought not to be a position of advantage in a truly collaborative organization, you’re thinking along the right lines. Having an *advantage* and collaboration are incompatible. By implication then, information hiding is incompatible with collaboration. And negotiation ought to be unnecessary.

Trust is the essence of a highly collaborative organization.

People can argue over the essence of agility or Lean Thinking but for me optimizing a software development organization for high performance starts with building a high trust organization. High trust organizations are flat. They feature a high degree of empowerment and delegation. They encourage joint responsibility and mutual

accountability. High trust organizations adapt dynamically to the needs of the organization regardless of reporting structure or formal organizational hierarchy. High trust organizations are social networks of highly collaborative knowledge workers.

High trust organizations are also lean. They expunge the overheads of low trust environments. High levels of trust and a flat structure mean that audits typical of hierarchies are eliminated and contracts are dispensed with. There is no concept of an internal market. In short, there is no negotiating. Negotiation is waste! The resultant contracts, their documentation, agreement, review and subsequent audit or enforcement are all considered as more waste! Subsequent negotiation for corrective action when required is yet more waste!

Negotiation is a symptom of an organization that has a lot of growth potential in its social capital. If you find yourself negotiating, you know there is room for improvement.

Be the Naked Organization

So if negotiation is undesirable, how do you go about eliminating it? Just saying “No!” won’t cut it. You need to get naked!

To put it another way, you need to unilaterally disarm! Stop hiding information. Make everything your team does transparently available to anyone who wants to take an interest. When everything is on show for all to see, you arrive at the negotiating table naked, exposed, no cards up your sleeve, nothing hidden - no sleeves. You have nothing to negotiate. All you can offer is your capability and availability.

Doing this is counter-intuitive. It goes against the grain of much of our education and training. But amazingly it is truly disarming. And it will change the culture of your whole business, not just technology. Rather than leave you *vulnerable to attack*, it generates trust in your relationships and it turns negotiation in to collaborative problem solving.

Think about it for a minute. “Vulnerable to attack.” From whom? Do you really think the marketing department wants to invade the IT shop and write the code themselves? Perhaps you think transparency leaves your team open to abuse from consumers of your services? Actually, the reverse is true. When you are negotiating from a position of hidden information, there is low trust. The consumer will believe you are hiding something which you are and will assume that he has to bargain hard. It’s likely that the consumer went to business school and you as an IT or software engineering manager did not. You are at a disadvantage. It is *lack of transparency* leaves you and your team open to abuse. Transparency protects you – despite the insecurity of nakedness and its associated vulnerability.

A Recipe for Success

So what is it that you are making transparently available to anyone who asks? You need to track the customer-valued work that you and your team does, from initial request to delivery. You need to report:

- how many things you've delivered
- how long they took to process (the lead time, from the customer's perspective)
- how many things you have in process
- the quality of your deliverables

In my Recipe for Success [3] Cutter Fellow Jim Highsmith discussed in a series of articles for Cutter's Agile Project Management advisory service [4-8], I identify four things your organization needs to do to deliver a high trust culture that is lean and exhibits agility. These are:

1. focus on quality
2. reduce (or limit) work-in-progress
3. balance demand against throughput
4. prioritize

Focusing on quality eliminates waste from rework, improves productivity and lead time, and builds trust with consumers.

Reducing work-in-progress sufficiently, allows team members to single-task. It reduces lead time and non-obviously has a huge positive effect on quality, further reducing lead time and improving productivity. It also forces the need for prioritization.

Balancing demand against throughput recognizes that a team can only effectively carry so much work-in-progress. Taking on too much clogs the system, affects quality and reduces customer service. Balance also eliminates abuse of the team - work/life balance or the agile idea of sustainable pace are achieved by agreeing to balance demand against capacity.

Finally, *prioritizing* forces the consumer to think hard about what is important. Given a supplier system that offers transparency on throughput, lead time, work-in-progress and quality, the consumer is left to figure out how best to utilize such a system for maximum value.

Collaboration Games

Transparency offers us the ability to turn negotiation in to collaborative problem solving. There is a simple question to be answered, "How best can we select job requests in order to maximize the value delivered through the supplier service?" Together your department and your value-chain partners can analyze and solve this problem. There is no negotiation. Negotiation is replaced by a puzzle of team optimization. Transparency, in this case, creates the opportunity for a collaboration game between consumer and supplier.

Naturally, there are a few snags. The work orders in a transparent system must be of a somewhat similar size. There must be a system of analysis that breaks work down in to types suitable for processing through a transparent system. If work items vary greatly in size, it leads to need negotiation. “If I give you two small ones, can you process them as if they were one regular item?” “How do I know they are two small ones?” Or, “We know this one is kind of big but it is really important, could you just squeeze it through?”

Soon you find yourself needing to estimate everything and to analyze the effort involved. Suddenly the problem to be solved revolves around trying to fit effort estimates against a calendar of available work hours. Since everyone knows that estimates are always wrong, and hence, the customer negotiator sharpens up her pencil and once again puts the squeeze on the supplier. I’ve recommended that organizations stop estimating [9] simply because it opens the door for abusive relationships through negotiation. Now I’m going a step further and asking you to stop negotiating. As part of that plan you need to stop estimating. Play with the facts! Use the hard, objective data, transparently. The hard facts are historical throughput (number of work orders delivered), lead time, quality, and quantity of work-in-progress. Estimates are not bad. They simply open the door to negotiation, reduce trust and leave, hard to build, social capital on the table.

It’s Non-negotiable: IT Services at Corbis

At Corbis, I’ve encouraged my organization to stop negotiating, to refuse to estimate, and to make all our data transparently available to anyone who wants it – often through popular software tools such as Microsoft Outlook or Sharepoint. In addition, we put things on whiteboards and post reports on walls.

We deal with different sized work items by offering different classes of service. For example, my build and configuration management team gets three different request types:

1. build requests
2. machine configuration and system maintenance requests
3. environment build-outs or replacements

Builds take tens of minutes, system maintenance can take up to a day and environment build outs can take a month or more. The department splits its capacity across the three classes of service. For example, we will agree to build out or refresh no more than one environment per month. This represents a specific resource allocation from the available pool of staff and time.

At a grander scale, we allocate 10% of our entire software engineering department resources to sustaining engineering, while we dedicate the other 90% to major initiatives. Our sustaining work takes work orders that represent 1 day to 15 days of coding effort and we offer a sustaining release to production every two weeks. We have analysis techniques in place to identify work orders that are too large and to either break them in

to smaller items that we process sequentially, or to reject the item and redirect it to a major initiative.

With major initiatives, we also process work orders – usually product features – and we batch these up for integration events every month and for release every 3 months.

Within our sustaining system, we offer several classes of service:

- change requests (new features requested by business owners)
- production bug fixes (escaped defects found in production)
- additional bug fixes (special class of service that utilizes slack developer capacity where testing is also performed by developers)
- maintenance work (IT system upgrades, patches, API refreshes and so forth)
- production data changes (refreshing meta-data for *data-driven* applications)

We use a kanban system [10] to control the resource allocation across the classes of service and to limit the work-in-progress in sustaining to the committed 10% of total resources.

The kanban system also facilitates prioritization. Each Monday morning at 10am, a prioritization board consisting of six vice presidents (or their representatives) from around the company works collaboratively to democratically select change requests or production bugs to fill available kanban slots. For instance, the board members may be told on Friday that there is one free kanban slot available. They are provided a list of the approved backlog of change requests and production bugs and given a simple instruction, “Please select one item.” At the Monday meeting, they get the opportunity to collaborate and select the optimal item to promote into the kanban slot for engineering to work.

This collaborative process emerged over several months. Initially, the board was in a bargaining mode. They would argue, “If I give you two small ones, can you treat it as only one slot?” but we would refuse to play that negotiating game. After two months, a “democratic” period emerged, in which the board self-organized into a democracy with multi-voting to select winners each week. However, after three more months, it was evident that democracy was not selecting the requests to maximize business value, and scarce capacity was being squandered. Out of this emerged the current collaboration game approach, in which the board members work together to puzzle over and select the best choice for the business regardless of functional boundaries or ownership of requests.

This is an example, of where the introduction of an IT process and its foundation in a culture of high trust and transparency, actually resulted in positive behavioral change across the wider business. Six previously competing business units have stopped negotiating and started collaborating and the result is a high performance organization that has released valuable new software to production every 9 business on average since September 2006.

Acknowledgements

This article was broadly inspired by an article in *Harvard Business Review*, by Philip Evans and Bob Wolf entitled “Collaboration Rules” [11]. Corey Ladas also contributed his thoughts and ideas inspiring the comparison of free market economics with those of collaborative networks.

References

- [1] Coase, Ronald H., *The Nature of the Firm*, Economica, 1937
- [2] Liebowitz, S. J., and Stephen E. Margolis, *Network Externalities (Effects)*, The New Palgrave's Dictionary of Economics and the Law, MacMillan, 1998
- [3] Anderson, David J., *Recipe for Success*, <http://www.agilemanagement.net/Articles/Weblog/RecipeForSuccess.html>, January 2007
- [4] Highsmith, Jim, *A Recipe for Success, Part 1*, Cutter Consortium, Agile Project Management Advisory Service E-mail Advisor, February 2007
- [5] Highsmith, Jim, *A Recipe for Success, Part 2*, Cutter Consortium, Agile Project Management Advisory Service E-mail Advisor, February 2007
- [6] Highsmith, Jim, *A Recipe for Success, Part 3*, Cutter Consortium, Agile Project Management Advisory Service E-mail Advisor, March 2007
- [7] Highsmith, Jim, *A Recipe for Success, Part 4*, Cutter Consortium, Agile Project Management Advisory Service E-mail Advisor, March 2007
- [8] Highsmith, Jim, *A Recipe for Success, Part 5*, Cutter Consortium, Agile Project Management Advisory Service E-mail Advisor, April 2007
- [9] Anderson, David J., *Stop Estimating*, <http://www.agilemanagement.net/Articles/Weblog/StopEstimating.html>, March 2005
- [10] Anderson, David J., *Kanban in Action*, <http://www.agilemanagement.net/Articles/Weblog/KanbaninAction.html>, March 2007
- [11] Evans, Philip and Bob Wolf, *Collaboration Rules*, Harvard Business Review, July-August 2005

Revision History

Revision 1.4, August 2007

About David J. Anderson

David Anderson is a thought leader in managing effective software teams. He is the President of Modus Cooperandi, a consulting firm dedicated to improving leadership in the IT and software development sectors.

He has 25 years experience in the software development business starting with computer games in the early 1980's. As a pioneer in the agile software movement David has managed teams at Sprint, Motorola and Corbis delivering superior productivity and quality. At Microsoft he developed the MSF for CMMI Process Improvement methodology.

David's book, **Agile Management for Software Engineering** – *Applying the Theory of Constraints for Business Results*, introduced many ideas from Lean and Theory of Constraints in to software engineering.

David was a founder and is a current board member of the APLN, a not for profit dedicated to promoting better standards of leadership and management in knowledge worker industries. He can be contacted at...

Email: dja@moduscooperandi.com